

Considerations about multistep community detection

Cristian Bisconti¹, Angelo Corallo¹, Laura Fortunato¹, and Antonio A. Gentile¹

Dept. of Innovation Engineering, University of Salento, 73100 Lecce, Italy,
 antonio.gentile@unisalento.it,
 WWW home page: <http://emi.unisalento.it/sna>

Abstract. The problem and implications of community detection in networks have raised a huge attention, for its important applications in both natural and social sciences. A number of algorithms has been developed to solve this problem, addressing either speed optimization or the quality of the partitions calculated. In this paper we propose a multi-step procedure bridging the fastest, but less accurate algorithms (coarse clustering), with the slowest, most effective ones (refinement). By adopting heuristic ranking of the nodes, and classifying a fraction of them as ‘critical’, a refinement step can be restricted to this subset of the network, thus saving computational time. Preliminary numerical results are discussed, showing improvement of the final partition.

Keywords: clustering, community detection, graph partitioning

1 Introduction

The framework of network analysis has proven a powerful tool in the study of complex phenomena, with applications ranging from biological and social systems, up to technological ones [1]. Gaining insight on the structure and behaviour of the network may often be considered a sort of *data fusion* problem, whenever huge data-sets are available.

Among the various strategies, outlined to understand large-scale structures, a successful one has pointed out the natural tendency of real-world networks to form *clusters*¹: groups of nodes densely connected among them, with sparser links to the rest of the network. Even though the concept is intuitively clear, an operational definition of a ‘network cluster’ is itself under debate: for a concise review of suggested definitions, see [3]. Identifying these dense structures inside a network may be crucial for a wide variety of reasons [4][5][6]. In other cases, it is already important the mere evaluation of the tendency to form clusters, without detecting cluster members. In fact, this tendency has been found indicative of robustness and stability of the network [7].

The importance of these applications has led recently to the intense development of algorithms, aiming to solve automatically the detection of communities, or to check for the clusterability of the network [2]. The focus is here on the specific case of *community detection*, where number and size of the clusters are free parameters of the problem [8], which addresses also the issue of determining if a *good* partitioning is achievable.

On a different basis, one could distinguish among classes of algorithms, grouped according to their focus. A first class, devoted to capturing the global picture of the network clustering, aiming at a fast solution of the clustering problem given, which especially suits large networks. Such algorithms will

¹ In the literature, for this same concept, also the following terms are equivalently used: *communities*, *groups*, *modules*, *partitions*. For slight preferences in the usage of these terms, refer to [2]

be generically indicated in the following as *coarse grain*, since in general they use global metrics as the figure of merit to optimize², and often embed approximated methods [11] [12], thus potentially leading to a relatively high rate of misclassified nodes (e.g. see [13]). On the opposite side, fine grain algorithms, in particular those involving metrics at the node/edge level³, or *hierarchical* structures: in this case, the aim is a precise assignment of the single nodes to the various communities. Moreover, these *refinement* algorithms frequently adopt ‘exact’ methods, for the optimization task they deploy.

The purpose of this paper is to provide a strategy, enabling to bridge these two different classes. At the moment, in fact, the norm is the straightforward application of a single step algorithm [2], or multi-step approaches with different optimization schedules for the same metric (see [15] for an example with *modularity*). There is a reason behind this tendency. Small networks can efficiently rely on time consuming algorithms, thus making superfluous to adopt faster methods. These last ones are instead the only feasible chance for large networks. In this paper, we envisage that it is possible to overcome this difficulty, by running a refinement step on only a fraction of the whole network. This fraction is identified via *heuristic metrics*: we call them ‘heuristic’ because, as better shown in the following, the metric chosen not only draws on the characteristics of the network analyzed, but must rely on some ‘preliminary’ clustering results, as computed via coarse algorithms. In Par.2, after a brief introduction on the framework of our proposal, we will provide a detailed assessment of general features and applicability of our multi-step scheme, and discuss a few metrics which may be adopted as heuristics. Characteristics and a first testing of the method, based on heuristics proposed, will be illustrated in Par.3. Some remarks and outlines of future developments conclude this work.

2 Framework and Methods

In the following, we are going to use concepts and metrics derived from graph theory, assuming that:

Proposition 1. *The network to analyze can be represented by a graph \mathcal{G} .*

For \mathcal{G} we adopt the following synthetic definition (see [16] for more details):

Definition 1. *A (directed) (weighted) graph is the ordered pair $\mathcal{G}(V, E)$, with V and E respectively the n vertices (v_i) and m edges belonging to G . If (directed), the edges $\{v_i, v_j\}$ are ordered pairs. The (weighted) values of the edges among vertices can be embedded in an ‘adjacency matrix’ A_{ij} of dimension n , where: $a_{ij} = 0$ iff there is no edge linking v_i to v_j .*

\mathcal{G} may be a *digraph*⁴. This case is explicitly analyzed in the following, where the ordering in the indexes of matrix A (in this case non-symmetric) will be supposed to follow the rule: edge $i \rightarrow j$ is embedded in the element a_{ij} , and viceversa. Also the case of a *multimodal graph* can be treated in principle, supposing that a clustering problem, as in Def.2, is well posed for the graph considered.

As pointed out in the introduction, in this paper we intend to address the general problem of automatic clustering in a network. Therefore, we will mainly refer to the case of:

² E.g. the *optimization methods* using: *E/I ratios* [9], information-compression measures [7], ..., Hamiltonian-like quantities (spin-hamiltonians [10], *modularity* [8], ...). Another good example is the class of methods known as *block modeling* [2].

³ Like the *edge betweenness*, *information centrality*, other cost functions, directly referred to the network structure (Kernighan-Lin approach [14], ...), or real-world analogies: *current-flow*, *message-passing*, ... [4]

⁴ For a detailed review of peculiarities of clustering approaches in directed graphs, the interested reader may refer to [17].

Definition 2. ‘Community detection’ as the optimal partition problem⁵ of finding \mathcal{C}_k non-overlapping, non-empty components (i.e. subgraphs) of \mathcal{G} : $\bigcup_k \mathcal{C}_k = \mathcal{G}$. Their number k may be an input of the problem, or left as a free parameter.

Prop.1 and Def.2 are strictly required, for the following discussion to make sense. The assumption of ‘no-overlap’, instead, may be (partially) relaxed, though leading to interesting applications. Moreover, if the partitions \mathcal{C}_k optimize a ‘quality function’⁶, it would be eased a quantitative comparison among different solutions (eventually found at different steps, or by different combinations, of the global scheme as in Fig.1). Further comments about these assumptions can be found in [18].

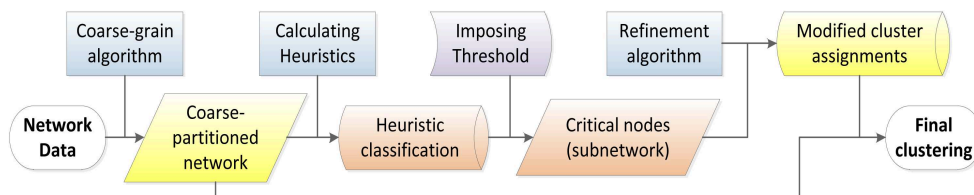


Fig.1: Flowchart of the multi-step method proposed in the text. In blue/violet are distinguished the operational steps. Other colors are referred to generic data (with database symbols) and data in graph-structure.

2.1 The multi-step scheme

Our contribution for a multi-step approach is the proposal of *heuristic metrics*, that have both low computational time-complexity, and a good efficiency in classifying the nodes according to their degree of membership to possible communities. These metrics enable the adoption of a scheme including:

1. a *coarse grain algorithm* for the initial clustering guess,
2. an efficient, *heuristic metric* for the retrieval of a reduced set of nodes, requiring further analysis upon cluster assignment,
3. a *refinement algorithm*, to be run on the nodes produced by the previous step, i.e. a fraction of the initial graph, to improve the ‘quality’ of the final partitions.

These three main elements, along with some other features which will be introduced in the text, are in Fig.1, which shows the global multi-step structure.

The very same introduction of a *refinement* brings along the problem of identifying a measure, able to compare different clustering solutions (i.e. a *relative* measure). A general discussion about the problem is clearly outside our scope: additional information can be found in [19],[20],[21]. In the following, the ultimate target will be to approximate those partitions, which would be provided if the fine-grain analysis chosen was to be performed on the whole network (implicitly assumed to be the best partitioning available). About the distance among partitions provided at different

⁵ We stress not to confuse it with the *graph partitioning*, i.e. a specific clustering problem, see [2]

⁶ Which may equivalently be a ‘cost function’: a survey of such functions is available in [2].

steps in our procedure, we adopt the ratio between the minimum number of elements to delete from a graph $D(P, P')$, so that the two induced partitions become identical, and the size of the graph [18]:

$$D(P, P') := n_D(P, P')/n \quad (1)$$

A multi-step procedure requires a few qualitative hypotheses, in order to be effective:

Proposition 2. *refinement algorithm used must be able to perform displacements of single nodes;*

Proposition 3. *the heuristic metric chosen must perform as a good figure of merit, in quantifying the ‘criticality’ of the nodes in the network;*

Proposition 4. *however chosen, the fastest method (eventually approximate) to compute the metric in Prop.3 should at least outperform, in time-complexity, the refinement clustering algorithm.*

Prop.2 derives from the necessity, once a node-ranking has been established, to analyze, and eventually modify, the cluster attribution of specific nodes, so to address the way the refinement algorithm works. The second statement emphasizes how a perfectly efficient metric should rank first *only* those nodes which will be misassigned by the coarse grain algorithm. Clearly, given that a variety of algorithms could be used as coarse-grain, this ‘perfect efficiency’ is indeed a relative concept, and independently from the refinement algorithm there is no way to define it. Finally, for an alternative clustering procedure to be competitive, with respect to the refinement algorithm, all of its steps must be (much) faster to compute, as stated in Prop.4.

A first naive approach, for retrieving the critical nodes of the network, could be to adopt *centrality* measures [22] from network theory. However, there are a few drawbacks [18], such as the implicit assumption, that the refinement should involve the most ‘important’ nodes. Misclassifying a central node is likely more problematic, but there is no general reason why the coarse-grain algorithm should perform worse on most central nodes.

Let us introduce a few qualitative statements, aiming to satisfy the requirements in Propp.2-4. As the first, the assignment of a node to a cluster depends the distribution of its links to neighbour nodes [2]: this leads to introducing the node degrees. In order not to relate the heuristic to the importance of the node, some normalization factor must be introduced. In undirected graphs, we will use a total ‘symmetrized’ degree for each node j , $d_T(j) := \sum_i (a_{ij} + a_{ji})/2$.

Given the hypothesis of computing heuristics only after a first coarse assignment of nodes to clusters, one is able to distinguish among edges *inside* or *outside* a given cluster, via the binary function com with values in $\{-1, 1\}$:

$$com(i, j) := \begin{cases} -1 & \text{(if } i \text{ and } j \text{ belong to different communities)} \\ +1 & \text{(if } i = j \vee \text{ if } i \text{ and } j \text{ belong to same cluster)} \end{cases} \quad (2)$$

We claim that a 1st order heuristic metric, suitable for quantifying the criticality of node j , can be formulated as:

$$H_1(j) = \frac{1}{2d_T(j)} \sum_i (a_{ij} + a_{ji}) com(i, j) \quad (3)$$

while for the 2nd order heuristic we suggest:

$$H_2(j) = \frac{1}{2d_T^2(j)} \sum_{i \neq j} (a_{ij} + a_{ji}) com(i, j) Q d_T(i) H_1(i) \quad (4)$$

where:

$$Q = \frac{\delta(\mathcal{G})}{\Delta(\mathcal{G})} \quad (5)$$

is a normalization factor, with $\delta(\mathcal{G})$ and $\Delta(\mathcal{G})$ the minimum and maximum degree of the nodes in \mathcal{G} , respectively.

A few remarks. The expressions about the *order* refer to the width of the network sample taken into account for each node: the edges shared *with* its neighbour nodes in the 1st case, and also all edges shared *by* its neighbour nodes in the 2nd.

Both heuristics are bounded: as it is easy to verify, $-1 \leq H_1, H_2 \leq +1$. Thus, the first order heuristic may be interpreted as a normalized measure of the *correlation* of the node with its cluster of assignment, disregarding its neighbour nodes. Evidently, a positive correlation is here an index of robust assignment, whereas negative correlations indicate misassignment.

Qualitatively, re-introducing in (4) the heuristic H_1 accounts for the cluster assignment of neighbour nodes: the stronger the connection of a neighbour node i to its own cluster, the higher we expect its contribution to the (mis)assignment score of analyzed node j , if $com(i, j) = +1$ (-1). The factor

$$M := Qd_T(i)/d_T(j), \quad (6)$$

instead, can be interpreted as a measure relating the contribution from node i to its relative ‘importance’ in the network, compared to node j (thus the presence of Q). That is, M reduces the contribution from H_2 , compared to H_1 : if $d_T(i)/d_T(j) = \rho \Rightarrow M < \rho^2$.

Another interesting point to analyze is how to combine the two heuristics introduced. We suggest that the most profitable figure of merit is the convex combination:

$$H(j) := \alpha H_1(j) + (2 - \alpha) H_2(j) \quad (7)$$

with $\alpha \in [0, 2]$. In the following, illustrating the proposal, we will restrict considerations to the simplest case with $\alpha = 1$.

It is worth to comment how the introduction of heuristics as above may be regarded as a ‘mean field like’ procedure, where only pairwise, nearest neighbour interactions are considered (which is the case, for example, in Ising models). The quantity H itself can be interpreted as a *potential*, once changed in sign. One may notice that procedures based on optimization of *Hamiltonians* have already been thoroughly applied to the clustering problem (e.g. [23] [24]). Indeed, with a terminology drawing on this parallel, a key difference in our approach is that we are defining and using *local* potentials, whereas the traditional approach involves the optimization of a *global* potential.

2.2 Further comments on the heuristics

Given that the heuristics, in the form introduced so far, were only intuitively justified to be reliable metrics for our aim, it is plenty of possible modifications, simplifying or generalizing the particular version given in (3) and (4).

We will take in consideration a few cases which may be interesting for some particular applications. As the first, whenever a speed-up in the computation of the heuristics is required, it is envisaged the possibility to slightly change the definition of *com* (2), so to skip operations on *positive* (or, equivalently, *negative*) terms. Therefore, this version of the algorithm could use e.g.:

$$com^+(i, j) = \begin{cases} 0 & \text{(if } i \text{ and } j \text{ belong to different communities)} \\ +1 & \text{(if } i = j \vee \text{ if } i \text{ and } j \text{ belong to same cluster)} \end{cases} \quad (8)$$

or viceversa for $com^-(i, j)$. Steps involving null terms in the computation of H_1 and H_2 would be excluded by conditional restraints.

A more interesting case is given by *directed* graphs (i.e. ‘digraphs’). In fact, to keep the general case as simple as possible, we have always avoided directionality considerations in (3) and (4), by using the averaged term $(a_{ij} + a_{ji})/2$. Intuitively, this is equivalent to the replacement of multiple directed (weighted) edges, for each couple of nodes, with a single undirected weighted edge. Even if approaches like this have been applied to highly successful analyses of naturally directed graphs [25], it is well recognized how intrinsic directional features may add insight to static [26] or dynamic [27] analyses of networks. Notice that the heuristics introduced may be readily generalized to include different expressions for an ‘in-metric’ $H_{1,2}^{in}$ as well as an ‘out-metric’ $H_{1,2}^{out}$. I.e. for the inner case:

$$H_1^{in}(j) = \frac{1}{d_T^{in}(j)} \sum_i a_{ij} com(i, j) \quad (9)$$

$$H_2^{in}(j) = \frac{1}{[d_T^{in}(j)]^2} \sum_{i \neq j} a_{ij} com(i, j) Q d_T^{in}(i) H_1^{in}(i) \quad (10)$$

Now, considerations about robustness of products of inner and outer quantities, for clustering procedures, may apply to this case. In fact, multiplying H_1^{in} and H_1^{out} , the product (H'_1) closely resembles⁷ the *vertex-cluster affinity*, employed in [28] for the *graph degree-linkage* method, where the cluster would here be the neighborhood \mathcal{N} of each critical vertex. The contribution from H'_2 can instead be seen as an improvement of this affinity. Therefore, drawing on these previous results, we claim that a robust implementation of our procedure in directional cases uses the node heuristics:

$$H'_1(j) := H_1^{in} H_1^{out} \quad (11)$$

$$H'_2(j) := H_2^{in} H_2^{out} \quad (12)$$

and the obvious generalization of (7) for their combination.

It is still left open, the possibility to drastically change the form of the heuristic metrics adopted. For example, given that H_1 is claimed to be a measure of the membership degree of node j to its initial community, one could recall how this indication is embedded in the elements of the *membership matrix*, as defined in [29]. However, the additional definitions of ‘positions’ and ‘distances’ in a metric space, required in the definition of this matrix, may be rather artificial for some graphs [2]. Again modifying preliminary definitions: Q , as given in (5), may be considered a rough figure of merit for the degree ratio in (6). E.g. one could assume a Gaussian behaviour in the degree distribution, and thus suppose Q to be in the form of a standard deviation⁸:

$$Q^2 = \frac{2}{n(n-1)} \sum_{\substack{i,j \\ i \neq j}} [\delta_D(i, j)]^2, \quad (13)$$

$$\text{with: } \delta_D(i, j) = (d_T(i) - d_T(j)) \quad (14)$$

⁷ The similarity of these quantities does not imply similarity in their usage, as in [28] the vertex-cluster affinity (and its derivatives) are directly used for the agglomerative step of the algorithm, whereas we use them only to classify the quality of single-node attributions to clusters.

⁸ Notice that the expected value for the population is trivially $\langle \delta_D \rangle = 0$.

and therefore it may be objected that:

$$M' := \exp(-\delta_D(i, j)^2/Q^2) \quad (15)$$

is a more reliable measure as a *degree distance* among nodes i and j and should replace M (6) in (4). Notice that (15) provides a measure, on the strength of the connection between the nodes, resembling the *dimensionality reduction* procedure invoked for graph construction in [30] and related works. However, two considerations hold. The first and more important is that computing the quantity in (13) is a computational problem much more costly (and in some cases even tricky [31]), than the linear scan required for computing (5). A second noticeable problem is that the naive introduction of this ‘standard variance’ form for Q does not fit well our requirements. Indeed, it is introduced an unwanted symmetry: M' is a factor reducing the importance of H_2 , indifferently of whose node is the degree centrality increasing (whereas in M this was true only in the situation $d_T(j) \gg d_T(i)$). In formulas, where ϵ is $O(e^{-n^2})$:

$$\lim_{d_T(j)/d_T(i) \rightarrow \infty} M \rightarrow 0 \quad (16)$$

$$\lim_{|d_T(j) - d_T(i)| \rightarrow \infty} M' \rightarrow \epsilon \quad (17)$$

There are certainly various possibilities to solve the issue: e.g. introducing further parameters in (14), or defining it differently. However, in our opinion this unnecessarily complicates the global picture, and therefore move on to test numerically the performance of the heuristics outlined.

	$H_1(i)$	$d_T(i)$	$com(i, j)$	Q	$\dots \sum_i (a_{ij} + a_{ji}) \dots$	$Total$
$H_1(i)$	-	$\mathcal{O}(m/n)$	c.g.?	-	$\mathcal{O}(m/n)$	$\mathcal{O}(m/n)$
$H_2(i)$	$\mathcal{O}(m/n)$	$\mathcal{O}(m/n)$	c.g.?	$\mathcal{O}(1)$	$\mathcal{O}(m/n)$	$\mathcal{O}(\frac{m}{n} + 1)$
size	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(n)$	$\mathcal{O}(1)$	$\mathcal{O}(m)$	$\mathcal{O}(m + n)$

Table 1: Analysis of computational complexity (average, per node) and memory usage (globally) of the quantities involved in the calculation of the heuristics in Eqs.(3) and (4), for an undirected graph. *c.g.?* indicates that the complexity of this step depends on the coarse algorithm applied. $com(i, j)$ is supposed to be retrieved from a stored vector of single-node assignments.

2.3 Discussion about implementation

We are now left with checking the responsiveness of our proposal for a heuristic, to the requirements stated in Pts. 2-3 of Prop.2. Observing Table 1, it is easy to see that H_1 has a complexity of $\mathcal{O}(m)$ and H_2 has a complexity of $\mathcal{O}(m + n)$, under the following assumptions: the graph is undirected and stored as an ordered *edgelist*⁹, coarse communities have already been calculated and stored in a vector. Notice how redundant terms in the two heuristics can ease the subsequent calculation of both quantities $H_{1,2}$. Such a complexity is a reasonably good result: one of the fastest coarse

⁹ If not, an additional step with complexity $\mathcal{O}(m \log m)$ must be taken into account

algorithms for community detection runs with complexity $\mathcal{O}(n+m)$ on sparse graphs. Additionally, operations leading to the heuristics' complexity are very basic, thus we envisage very low factors.

In order to perform a test for the multi-step scheme, following also Fig.1, two elements are required to be explicated.

A *coarse grain algorithm* for the first step. We chose to use the *fast Newman* (FN) approach [32] with a greedy modularity optimization of the *modularity*, as suggested in [11]. Within this implementation, it is known to run in $\mathcal{O}(n \log^2 n)$ on sparse graphs. This method is of widespread adoption in the literature¹⁰ and in several network analysis softwares.

A *refinement algorithm* for the final step. In this case we introduced a modified *Girvan-Newman* (GN) method, based on the *edge betweenness*: a perfect example of an algorithm unfeasible to be used straightforward for large networks, as it requires $\mathcal{O}(n^3)$ time (sparse case). The original version of this algorithm was not intended to perform single node re-assignments [34], so that it is here modified, even though keeping the same local measure as the working principle. In brief, here the edge betweenness is calculated only for *critical edges*, i.e. those edges linking couples of nodes, of whose at least one is *critical*. The last edge to be removed, before a node is isolated, is also the one ruling the community assignment¹¹. Notice that the refinement algorithm used is allowed both to eventually shrink the number of clusters composing the final partitioning, *and* to create new clusters, eventually not resolved by the coarse step.

The adoption of a refinement step poses a non-trivial problem: given unawareness of the percentage of nodes classified in the wrong cluster by the coarse algorithm, how many nodes must be 'refined' analyzed, among those scoring worse in H ? That is, we need to impose a threshold to the heuristics (see Fig.1), selecting as critical nodes only those having a lower value of H . In our opinion, this point requires a good insight about the structure of the network, and if investigated, can provide interesting results. A pragmatic and prudential solution is: pose the threshold in H as high, as the additional computational time, required for refinement, is considered feasible by the adopter. For numerical tests below, we will adopt instead an 'absolute' approach: the refinement algorithm will be run on all, and only, those nodes having negative values of H .

3 Preliminary tests

This paragraph is devoted to show how the particular implementation of a multi-step scheme (as outlined in Par.2.3) works for a real case, and in particular to test if the heuristics, introduced so far, are capable of satisfying the requirements stated at the beginning of this section.

Test-cases. We have chosen to focus on the split of a *karate club* in two different 'communities', studied in [35]. This example fits well a preliminary, qualitative discussion, because it is small enough ($n=34$) to let us follow in detail the performance of the heuristics, yet it is complex enough to pose difficulties for the fast coarse algorithm chosen [8]. In Fig.2 we plot the sum of the heuristics H_1

¹⁰ Its combined simplicity and robustness make the FN method still very popular, even if several works have started to point out its ineffectiveness for specific cases [13], [33].

¹¹ Specifically, we progressively remove critical edges with high edge-betweenness. The algorithm has three hierarchical rules to assign node to the refined *community vector*: i) if a critical node i is left with one only edge linking it to a non-critical node j (i.e. $a_{ij} + a_{ji} \neq 0$), i acquires the same community assignment of j : $i \in \mathcal{C}_k$ iff $j \in \mathcal{C}_k$; ii) for critical nodes pointing to each other, before becoming isolated by the edge-removal procedure ('queued nodes'), it is attempted the creation of a new community; iii) if this attempt fails, the *transitivity principle* introduced in the text is used to infer the non-critical node ruling the community assignment. The full algorithm in pseudocode can be found in the Appendix.

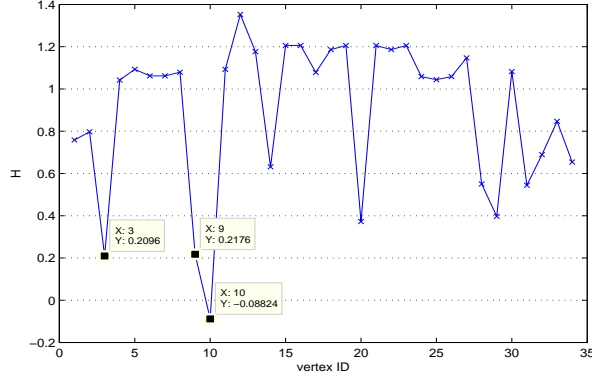


Fig. 2: Combined heuristic H (Eq.7) for the Zachary’s karate club case. Datatips are displayed in the first case for those vertices exhibiting the lower scoring, and therefore the most critical ones. Vertices IDs are the same as in [34].

and H_2 for all the vertices of the karate club network, after a coarse assignment of clusters has been performed through the application of the FN algorithm. It is immediately evident how almost all of the nodes have positive values of both $H_{1,2}$. This confirms that H captures the good performance of the FN algorithm in this test. We can also state that our core claim is satisfied: the node #10, known to be misclassified by the coarse algorithm [32], is the one scoring worse, and even has a negative H , as shown in Fig.2. Notice also that the GN refinement correctly classifies this node, displacing it into the ‘right’ cluster. Recalling (1), the coarse method has in this case a distance of $D \cong 0.029$ from the partition found by our scheme: this distance can be understood as the *improvement* provided for the solution. Noticeably, for this specific case, the GN method is known to classify incorrectly node #3 [34], which actually ranked worst, immediately after node #10, in the H scoring. This further suggests how the heuristic proposed is indeed efficient, in sorting nodes with uncertain cluster assignment.

As a counter-example, in Fig.3 we report the heuristics, calculated after the same coarse step, run on an artificial network¹² with $n=128$. This network resembles a ‘Girvan-Newman benchmark’ with communities of variable size, and with parameters which are known to make the community assignment fail, when performed by the FN algorithm [36][37]. It is immediately evident how the average scoring of the heuristic H is much worse than the previous case, and how most of the nodes exhibit negative scoring. This indicates again that the heuristics scouts nodes misclassified in the coarse step.

4 Conclusions

Summarizing the main results of this work: we have proposed the adoption of a multi-step scheme, to improve the results of clustering algorithms, with a particular focus on community detection. This scheme basically includes: the adoption of a (state-of-art) fast, coarse algorithm for the first step; an

¹² Created through the benchmark package available at: goo.gl/Btp70b

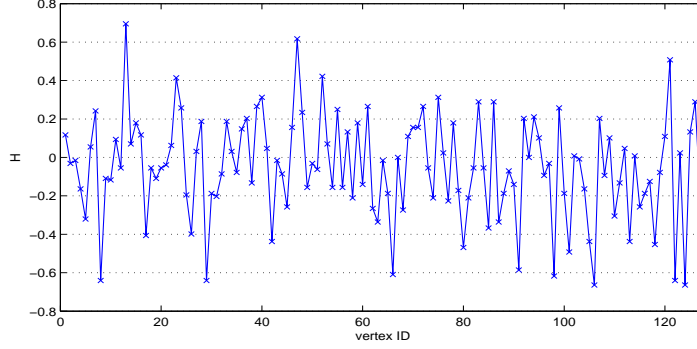


Fig. 3: Combined heuristic H for the *Girvan-Newman benchmark* cited in the text. Generated using the code as in [37], with average degree $\bar{d} = 16$, and mixing parameter $\mu = 0.6$.

accurate refinement algorithm, specifically adapted for this purpose; to bridge these two elements, a novel set of heuristic metrics. These last ones are the core of the proposal: they are intended to scout those nodes potentially tricky in the cluster assignment, and thus worth to be analyzed by the refinement step. We have shown, with the aid of test-cases, that the heuristic introduced satisfies the requirements of being computable with low time-complexity, and may efficiently retrieve those nodes which turn out to ‘deceive’ the less accurate algorithms.

In future developments there is the plan to systematically investigate to what extent our approach reveals useful for application to real world and computer generated networks (thus identifying its limits). In particular, the aim will be about large scale networks, for which it may also be unknown the ‘true partitioning’ (whether obtained via a direct observation, or as provided by the application of the refinement to the whole network). In this case the only possible check would be the comparison with results, as provided by different fast algorithms. Another direction, for further analyses, is given by the limitations already found for modularity-based approaches [13]: we claim that our multi-step strategy may (partially) solve the degeneracies displayed by these approaches for particular cases. Verification of this conjecture could lead to important applications.

Appendix A

It is reported below the pseudocode related to the *GN refinement* used for numeric experiments in the text (Par.3).

```

program 'GN refinement' (coarse com-vector){
  for (each node in graph){
    if (H (node) > threshold-1){
      assign (node to critical-nodes)
      assign (edges connecting the node to critical-edges)}}
  do edge_betweenness (critical-edges)
  sort (critical-edges, descending edge_betweenness)
  while (there are critical-edges left){
    newgraph = remove (critical-edge with highest betweenness from graph)
    if (critical-edge connects two nodes, of which one  $\notin$  critical-nodes){
      header-node[critical-node]= its neighbour-node from removed edge
      assign increasing priority[critical-node]}
    find (critical-nodes left with one only edge)
  }
}

```

```

for (each of these nodes){
  if (the only neighbour-node left  $\in$  critical-nodes){
    assign (node to queued-nodes)
    follower-node[neighbour-node]=node}
  else{
    com-vector[node]=com-vector[neighbour-node]
    assign(node to solved-nodes)}}}
listed-nodes = merge (queued-nodes with corresponding neighbour-nodes)
extract (from graph the subgraph including all and only the listed-nodes)
comps = connected_components (subgraph)
for (each comp){
  if (sizeof comp > threshold-2){
    assign(nodes in comp to novel community)
    remove(queued nodes in comp from queued-nodes)}}
for (each of the remaining queued-nodes){
  if (neighbour-node[queued-node]  $\in$  solved-nodes){
    com-vector[queued-node]=com-vector[neighbour-node[queued-node]]
    assign (queued-node to solved-nodes) & remove(node from queued-nodes)}}
  while (there are queued-nodes left){
    move to next(header-node with highest priority, connecting one queued-node)
    com-vector[corresponding queued-node]=com-vector[its header-node]
    assign (queued-node to solved-nodes) & remove(node from queued-nodes)
    for (all follower-nodes[this solved-node]){
      com-vector[follower-node]=com-vector[solved-node]
      assign (follower-node to solved-nodes) & remove(node from queued-nodes)}}
return(new com-vector)}}

```

References

1. U. Brandes and T. Erlebach, *Network analysis: methodological foundations*, vol. 3418. Springer, 2005.
2. S. Fortunato, “Community detection in graphs,” *Physics Reports*, vol. 486, no. 3, pp. 75–174, 2010.
3. S. Fortunato, V. Latora, and M. Marchiori, “Method to find community structures based on information centrality,” *Physical Review E*, vol. 70, no. 5, p. 056104, 2004.
4. M. Newman, “Communities, modules and large-scale structure in networks,” *Nature Physics*, vol. 8, no. 1, pp. 25–31, 2011.
5. H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási, “The large-scale organization of metabolic networks,” *Nature*, vol. 407, no. 6804, pp. 651–654, 2000.
6. A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig, “Syntactic clustering of the web,” *Computer Networks and ISDN Systems*, vol. 29, no. 8, pp. 1157–1166, 1997.
7. M. Rosvall and C. T. Bergstrom, “An information-theoretic framework for resolving community structure in complex networks,” *Proceedings of the National Academy of Sciences*, vol. 104, no. 18, pp. 7327–7331, 2007.
8. M. E. Newman, “Modularity and community structure in networks,” *Proceedings of the National Academy of Sciences*, vol. 103, no. 23, pp. 8577–8582, 2006.
9. D. Krackhardt and R. N. Stern, “Informal networks and organizational crises: An experimental simulation,” *Social psychology quarterly*, pp. 123–140, 1988.
10. P. Ronhovde and Z. Nussinov, “Multiresolution community detection for megascale networks by information-based replica correlations,” *Physical Review E*, vol. 80, no. 1, p. 016109, 2009.
11. A. Clauset, M. E. Newman, and C. Moore, “Finding community structure in very large networks,” *Physical review E*, vol. 70, no. 6, p. 066111, 2004.
12. V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, “Fast unfolding of communities in large networks,” *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2008, no. 10, p. P10008, 2008.
13. B. H. Good, Y.-A. de Montjoye, and A. Clauset, “Performance of modularity maximization in practical contexts,” *Physical Review E*, vol. 81, no. 4, p. 046106, 2010.
14. B. Kernighan and S. Lin, “An efficient heuristic procedure for partitioning graphs,” *Bell system technical journal*, 1970.

15. M. E. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical Review E*, vol. 74, no. 3, p. 036104, 2006.
16. D. B. West *et al.*, *Introduction to graph theory*, vol. 2. Prentice hall Englewood Cliffs, 2001.
17. F. D. Malliaros and M. Vazirgiannis, "Clustering and community detection in directed networks: A survey," *Physics Reports*, vol. 533, no. 4, pp. 95–142, 2013.
18. A. A. Gentile, A. Corallo, C. Bisconti, and L. Fortunato, "Proposal for heuristics-based refinement in clustering problems," in *Proceedings of the SOCNET '14 Workshop (to be published)*, University of Bamberg, 2014.
19. R. Kannan, S. Vempala, and A. Vetta, "On clusterings: Good, bad and spectral," *Journal of the ACM (JACM)*, vol. 51, no. 3, pp. 497–515, 2004.
20. H.-P. Kriegel and M. Pfeifle, "Measuring the quality of approximated clusterings," in *BTW*, vol. 5, pp. 415–424, 2005.
21. C. Robardet, F. Feschet, and N. Nicoloyannis, "An experimental study of partition quality indices in clustering," in *Proceedings of the 4th European Conference on Principles of Data Mining and Knowledge Discovery, PKDD '00*, (London, UK), pp. 599–604, Springer-Verlag, 2000.
22. S. Wasserman, *Social network analysis: Methods and applications*, vol. 8. Cambridge university press, 1994.
23. J. Reichardt and S. Bornholdt, "Detecting fuzzy community structures in complex networks with a potts model," *Physical Review Letters*, vol. 93, no. 21, p. 218701, 2004.
24. S. Liu, L. Ying, and S. Shakkottai, "Influence maximization in social networks: An ising-model-based approach," in *Communication, Control, and Computing (Allerton), 2010 48th Annual Allerton Conference on*, pp. 570–576, IEEE, 2010.
25. A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, 1999.
26. A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener, "Graph structure in the web," *Computer networks*, vol. 33, no. 1, pp. 309–320, 2000.
27. P. Krapivsky, G. Rodgers, and S. Redner, "Degree distributions of growing networks," *Physical Review Letters*, vol. 86, no. 23, p. 5401, 2001.
28. W. Zhang, X. Wang, D. Zhao, and X. Tang, "Graph degree linkage: agglomerative clustering on a directed graph," in *Computer Vision–ECCV 2012*, pp. 428–441, Springer, 2012.
29. J. C. Bezdek, *Pattern recognition with fuzzy objective function algorithms*. Kluwer Academic Publishers, 1981.
30. M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation*, vol. 15, no. 6, pp. 1373–1396, 2003.
31. T. F. Chan, G. H. Golub, and R. J. LeVeque, "Algorithms for computing the sample variance: Analysis and recommendations," *The American Statistician*, vol. 37, no. 3, pp. 242–247, 1983.
32. M. E. Newman, "Fast algorithm for detecting community structure in networks," *Physical Review E*, vol. 69, no. 6, p. 066133, 2004.
33. A. Kehagias, "Bad communities with high modularity," *preprint arXiv:1209.2678*, 2012.
34. M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.
35. W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of anthropological research*, pp. 452–473, 1977.
36. L. Danon, A. Díaz-Guilera, and A. Arenas, "The effect of size heterogeneity on community identification in complex networks," *Journal of Statistical Mechanics: Theory and Experiment*, vol. 2006, no. 11, p. P11010, 2006.
37. A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical Review E*, vol. 78, no. 4, p. 046110, 2008.